Variable Neighborhood Search for the Degree-Constrained Minimum Spanning Tree Problem

Celso C. Ribeiro
Department of Computer Science
Catholic University of Rio de Janeiro
Brazil

Maurício C. de Souza
LIMOS, Université Blaise Pascal
France

Summary

1. Problem Formulation
2. Variable Neighborhood Search (VNS)
3. Local Search: Variable Neighborhood Descent (VND)
4. Computational Results
5. Concluding Remarks

Definitions

- $G = (V, E)$ connected undirected graph
- $w : E \to \mathbb{R}_+$ non-negative weights
- $b : V \to \mathbb{N}$ non-negative integer valued degree bounds
- d_i : degree of node i in a spanning tree of G

Degree-Constrained Minimum Spanning Tree Problem (DCMST): Find a minimum spanning tree of G with $d_i \leq b_i \; \forall i \in V$

- NP-hard
- Applications: design of circuits and communication networks
- Exact algorithms: Narula and Ho [1980], Savelsbergh and Volgenant [1985], Volgenant [1989], Gavish [1982]
- Heuristics: Narula and Ho [1980], Zhou and Gen [1997], Craig, Krishnamoorthy, and Palaniswami [1996], Krishnamoorthy, Ernst, and Sharaiha [1997]

Variable Neighborhood Search

- Mladenović and Hansen [1996]
- T : spanning tree of G satisfying all degree constraints
- $N_{vns}^{(k)}$: set of all trees T' such that:

  T' is feasible ($d_i \leq b_i \quad \forall i \in V$)

  T and T' have at least $|V| - 1 - k$ edges in common
- Neighborhood structure for DCMST:

  sequence of neighborhoods $N_{vns}^{(1)}, \ldots, N_{vns}^{(k_{max})}$

  $k_{max} \leq |V| - 1$
- VNS explores neighborhoods of increasing order
- Associated to each VNS iteration:

  current solution T

  neighborhood $N_{vns}^{(k)}$ of order k

Variable Neighborhood Search

- Each VNS iteration has two main steps:

  T' is randomly generated within $N_{vns}^{(k)}$

  local search is applied to T'

- If an improving solution is found, then:

  update the current solution

  resume the search using $N_{vns}^{(1)}$

- Otherwise:

  resume the search using $N_{vns}^{(k+1)}$

- Stopping criterion: traverse a sequence of *seqs* times the sequence $N_{vns}^{(1)}, \ldots, N_{vns}^{(k_{max})}$ of neighborhoods

Variable Neighborhood Search

VNS heuristic for DCMST:

procedure `VNS_DCMST`
  Let T_0 be an initial solution
  $T \leftarrow T_0$
  Set k_{max} to some value in $\{2, \ldots, |V| - 1\}$
  for $i = 1, \ldots, seqs$ **do**
    $k \leftarrow 1$
    while $k \leq k_{\max}$ **do**
      Randomly generate $T' \in N_{vns}^{(k)}(T)$
      Obtain $\bar{T}$ by applying a local search to T'
      if $w(\bar{T}) < w(T)$
      then $T \leftarrow \bar{T}$; $k \leftarrow 1$
      else $k \leftarrow k + 1$
      end-if
    end-while
  end-for
  return T
end-procedure

Initial Solutions

- Randomized construction
- G : complete graph with $b_i \geq 2 \quad \forall i \in V$
- Start with a forest of isolated nodes
- A randomly selected edge is inserted if:

  it connects two different components of the forest

  it does not violate the degree constraints

Local Search

- Start from feasible solution T'
- Edge-exchange operation defined by a pair (u, v) of edges

  u : edge in T' (to be removed)

  v : edge not in T' (to be inserted)
- Resulting graph is still a spanning tree of G
- Neighborhood definitions:

  $N^{(1)}_{vnd}$, $N^{(2)}_{vnd}$, and $N^{(3)}_{vnd}$

  (resp. one, two, and three edge-exchanges)
- Edges not in T' divided into three sets:

  E_0 : edges whose both extremities are not saturated

  E_1 : edges with exactly one extremity saturated

  E_2 : edges whose both extremities are saturated

Neighborhood $N_{vnd}^{(1)}$

Edge-exchange operations (u, v) which preserve feasibility:

- $v \in E_0$, maximum weight edge u in the cycle
- $v \in E_1$, adjacent edge u to the saturated extremity of v in the cycle

Move value:

- $w_v - w_u$

Number of moves:

- $O(|E_0| + |E_1|)$

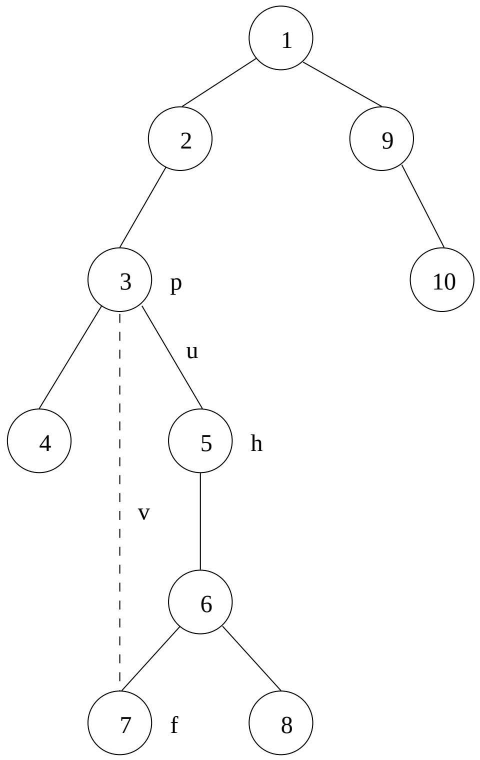
Complexity of each move evaluation:

- $v \in E_0$: linear time in the number of edges in the cycle
- $v \in E_1$: linear time in the degree of the saturated extremity of v

Neighborhood $N_{vnd}^{(1)}$

Example:

- $v = (p, f) \in E_1$
- $b_i = 3 \quad \forall i \in V$
- extremity p saturated
  extremity f not saturated
- edge $u = (p, h)$ adjacent to p in the cycle



Neighborhood $N_{vnd}^{(2)}$

Two edge-exchange operations (u_1, v_1) and (u_2, v_2):

- First operation leads to a tree with exactly one vertex violating the degree constraints

  $v_1 \in E_1$, maximum weight edge u_1 in the cycle

  $v_1 \in E_2$, maximum weight edge u_1 among the two in the cycle which are adjacent to the extremities of v_1

- Second operation restores feasibility

  $\hat{T}$: infeasible tree obtained after the first operation

  p : unique vertex violating degree constraints in $\hat{T}$

  u_2 : adjacent edge to p

  $v_2 \in E_0 \cup E_1$ connecting components created in $\hat{T}$

Move value: $w_{v_1} + w_{v_2} - w_{u_1} - w_{u_2}$

Number of moves: $O((|E_1| + |E_2|) \times (|E_0| + |E_1|))$

Complexity:

- First operation: linear time in the number of edges in the cycle

- Second exchange operation requires a depth-first search traversal of G: $O(|V| + |E|)$

Neighborhood $N_{vnd}^{(2)}$

Example:

- $b_i = 3$ for all $i \in V$

Neighborhood $N_{vnd}^{(3)}$

Three exchange operations (u_1, v_1), (u_2, v_2), and (u_3, v_3):

- First operation leads to a tree with exactly two vertices violating the degree constraints

  $v_1 \in E_2$, maximum weight edge u_1 in the cycle

- Second and third exchange operations restore feasibility

Move value: $w_{v_1} + w_{v_2} + w_{v_3} - w_{u_1} - w_{u_2} - w_{u_3}$

Number of moves:

- $O(|E_2| \times |E_0 \cup E_1|^2)$
- $O(|E_2| \times |E_0 \cup E_1|)$ moves effectively evaluated

Complexity:

- First operation: linear time in the number of edges in the cycle
- Second and third exchange operations require a depth-first search traversal of G: $O(|V| + |E|)$

Variable Neighborhood Descent

- Mladenović and Hansen [1996]
- Iterative improvement starting with $N_{vnd}^{(1)}$
- Local optimum: current neighborhood is changed
- Improving move: return to the first neighborhood
- Local optimum with last neighborhood: algorithm stops

procedure `VND_DCMST`(T')
    $l \leftarrow 1$
    while $l \leq 3$ **do**
        Find the best solution $T'' \in N_{vnd}^{(l)}(T')$
        if $w(T'') < w(T')$
        then $T' \leftarrow T''$; $l \leftarrow 1$
        else $l \leftarrow l + 1$
        end-if
    end-while
    $\overline{T} \leftarrow T'$
    return $\overline{T}$
end-procedure

Computational Experiments

- Implementation in `C` (AIX `gcc` compiler version 2.8.1)
- 200 MHz RISC 6000 workstation with 64 Mb of RAM

Test problems ZG:

- Genetic algorithm: Zhou and Gen [1997]
- Five problems in complete graphs for $|V| = 10, \ldots, 50$
- Edge weights uniformly distributed in [10,100]
- Lower bound: weight of the minimum spanning tree

Test problems and results from Krishnamoorthy et al [1997]:

- 200 MHz DEC Alpha processor
- Series CRD, SYM, STR, SHRD
- CRD and SYM: random Euclidean problems, complete graphs, $|V| \leq 100$, $b_i = 3$, ten instances for each size
- SHRD: 24 more difficult problems with $b_i = 2, 3, 4$ and $|V| = 15, 20, 25, 30$, two instances for each combination
- STR: 90 difficult Euclidean instances, nodes in clusters, $|V| = 30, 50, 70, 100$, $b_i = 3, 4, 5$, two instances for each combination

Computational Experiments

- Parameter tuning using 24 SHRD instances:

    Highest order neighborhood: $k_{max} = 0.4 \times (|V| - 1)$

    Stopping criterion: $seqs = 1$

- Four algorithmic strategies compared:

    Randomly constructed initial solution

    Local search applied to initial solution using $N_{vnd}^{(1)}$

    VND applied to the initial solution

    Full VNS/VND with above parameter values

Instances SHRD	Start	LS	VND	VNS
average relative error (%)	228.42	4.79	0.27	0.00
best solutions found	0	0	9	23
average time (seconds)	0.00	0.00	0.04	0.21

VND significantly improves upon LS

VNS found 23 out of 24 best solutions

Computational Experiments

Comparison with Zhou and Gen's genetic algorithm:

Problem				GA		VNS		
Instance	$\|V\|$	b_i	LB	value	deviation (%)	value	deviation (%)	sec's
ZG1	10	3	117	123	5.13	123	5.13	0.00
ZG2	20	3	233	237	1.72	233	0.00	0.00
ZG3	30	3	316	327	4.48	317	0.32	0.03
ZG4	40	3	419	449	7.16	420	0.24	0.09
ZG5	50	3	513	554	7.99	514	0.19	0.22

LB: minimum spanning tree

Better solutions for 4 out of 5 instances (same solution for remaining instance)

Computational Experiments

Comparison with Krishnamoorthy, Ernst, and Sharaiha's heuristic:

Problem			PSS		VNS	
Class	$\|V\|$	b_i	deviation (%)	sec's	deviation (%)	sec's
CRD	30	3	0.04	5.80	0.00	0.02
CRD	50	3	0.04	18.74	0.00	0.13
CRD	70	3	0.00	40.06	0.00	0.52
CRD	100	3	0.00	87.33	0.00	2.46
SYM	30	3	0.04	9.34	0.02	0.03
SYM	50	3	0.30	45.01	0.00	0.18
SYM	70	3	2.53	99.34	0.00	0.60

KES report best results with Problem Space Search (PSS)

Smaller average deviations for every class in smaller computation times

69 optimal solutions out of 70 instances (remaining instance: 0.2% deviation)

Computational Experiments

Comparison with Krishnamoorthy, Ernst, and Sharaiha's heuristic: SHRD instances (1)

File	$\|V\|$	b_i	Best	VNS	sec's
SHRD150	15	3	582	582	0.03
SHRD150	15	4	433	* 430	0.02
SHRD150	15	5	339	339	0.01
SHRD159	15	3	598	* 597	0.01
SHRD159	15	4	430	430	0.02
SHRD159	15	5	332	332	0.02
SHRD200	20	3	1105	* 1088	0.12
SHRD200	20	4	818	* 802	0.06
SHRD200	20	5	632	* 627	0.09
SHRD209	20	3	1114	* 1092	0.04
SHRD209	20	4	808	* 799	0.06
SHRD209	20	5	630	* 629	0.03
SHRD258	25	3	1786	* 1745	0.25
SHRD258	25	4	1290	* 1277	0.30
SHRD258	25	5	1007	* 999	0.12

Computational Experiments

Comparison with Krishnamoorthy, Ernst, and Sharaiha's heuristic: SHRD instances (2)

File	$\|V\|$	b_i	Best	VNS	sec's
SHRD259	25	3	1792	* 1756	0.20
SHRD259	25	4	1309	* 1292	0.15
SHRD259	25	5	1025	* 1016	0.22
SHRD300	30	3	2645	* 2595	0.47
SHRD300	30	4	1937	* 1905	0.37
SHRD300	30	5	1526	* 1504	0.32
SHRD309	30	3	2621	* 2585	0.71
SHRD309	30	4	1911	* 1898	0.78
SHRD309	30	5	1486	* 1474	0.84

Improved best known solutions for 20 out of 24 instances (same solution quality for remaining instances)

Average improvement with respect to the best known solutions over the 24 instances: 1.02%

Small computation times

Computational Experiments

Comparison with Krishnamoorthy, Ernst, and Sharaiha's heuristic: STR instances (1)

Problem			GA-F		VNS	
$\lvert V\rvert$	b_i	dim.	error (%)	sec's	error (%)	sec's
50	4	3	0.00	24.34	0.00	0.32
50	4	4	0.00	23.04	0.00	0.39
50	4	5	0.72	24.11	0.00	0.46
50	4	6	0.15	22.57	0.00	0.51
50	4	7	0.57	23.66	0.00	0.54
50	5	3	0.03	19.87	0.00	0.24
50	5	4	0.00	19.60	0.00	0.35
50	5	5	0.54	20.00	0.00	0.40
50	5	6	0.07	19.48	0.00	0.49
50	5	7	0.32	19.70	0.00	0.51

KES report best results with a genetic algorithm

Class for which VNS achieved better improvements: optimal or best known solutions for all instances

VNS found better solutions than GA-F for 35 out of 45 pairs of instances (same solution quality for remaining instances)

Small computation times

Computational Experiments

Comparison with Krishnamoorthy, Ernst, and Sharaiha's heuristic: STR instances (2)

Problem			GA-F		VNS	
$\|V\|$	b_i	dim.	error (%)	sec's	error (%)	sec's
100	3	3	0.17	257.55	0.00	7.58
100	3	4	1.10	271.85	§ -0.01	7.39
100	3	5	0.07	314.29	0.00	7.09
100	3	6	0.58	203.85	§ -0.01	11.52
100	3	7	0.58	233.01	0.00	8.88
100	4	3	0.00	245.80	0.00	5.55
100	4	4	0.38	240.35	0.00	6.79
100	4	5	0.02	254.46	0.00	7.55
100	4	6	0.51	233.18	0.00	8.62
100	4	7	0.51	232.48	0.00	9.15
100	5	3	0.00	220.06	0.00	4.14
100	5	4	0.05	213.79	0.00	6.45
100	5	5	0.00	219.51	0.00	7.08
100	5	6	0.46	206.02	0.00	7.73
100	5	7	0.48	207.28	0.00	8.83

Concluding Remarks

- VNS-VND approach for the degree-constrained minimum spanning tree problem
- VNS and VND use different neighborhood structures
- Computational experiments on literature problems
- VNS-VND found better results than other heuristics within very small computation times
- Both VNS and VND lead to improvements in solution quality
- Extensions:
  - Computational experiments on broader and larger test problems
  - Improved initial solution construction procedures
  - Hybridization with GRASP
  - Extension to the capacitated minimum spanning tree problem